INF122 – Obligatorisk oppgave 1, Høst 2021

I denne oppgaven er det ikke lov til å bruke funksjoner fra andre moduler enn Prelude (i.e. du kan ikke importere andre pakker, f.eks. Data.List, og bruke funksjoner derifra for å løse oppgavene). Det er heller ikke lov til å levere felles løsninger. Du kan samarbeide med medstudenter for å komme frem til løsninger, men koden må skrives på egen hånd. Kopier av kode (fra hverandre eller andre kilder) aksepteres ikke. En kjørbar fil med løsning skal leveres elektronisk senest: **ONSDAG, 22. september, kl 23:59.**

I hver deloppgave blir du bedt om å programmere en funksjon. **Du kan ikke endre navn eller type på funksjonene!** Men du kan legge til flere hjelpefunksjoner der du selv kan bestemme typer.

Introduksjon og overblikk

Året er 1984 og du har nettopp begynt i en ny jobb for Sannhetsministeriet. Her har du fått i oppgave å oversette tekster fra «Oldspeak» (vanlig engelsk) til «Newspeak» (Partiets nye språk). For å imponere de nye sjefene dine, og ikke minst Storebror, har du bestemt deg for å effektivisere prosessen med Haskell-kunnskapene dine. Du har en liste med Newspeak-ord og deres oversettelse til Oldspeak. Du vil lage et program som går gjennom en tekst, finner alle Oldspeak-ordene og bytter disse ut med tilsvarende Newspeak-ord. Ord som finnes i listen, men ikke har en oversettelse skal fjernes og hver karakter i ordet erstattes med '*'.

1) isPrefix :: String -> String -> Bool

Programmer funksjonen **isPrefix :: String -> String -> Bool**. Denne funksjonen skal sjekke om det første parameteret er en prefix av det andre parameteret (dvs det andre parameteret starter med det første parameteret).

Eg:
isPrefix "foo" "foobar" = True
isPrefix "bar" "foobar" = False
isPrefix "" "foobar" = True

2) locate :: String -> String -> [(Int,Int)]

Programmer funksjonen **locate :: String -> String -> [(Int,Int)]**. Funksjonen skal finne alle forekomstene av det første parameteret i det andre parameteret og returnere startindeks og sluttindeks+1 til hver av forekomstene.

Eg:
locate “oo” “foobarbarfoo” = [(1,3), (10,12)]
locate “oo” “barbarbar” = []
locate “” “foobar” = []

(Hint: her kan det være lurt med en hjelpefunksjon slik at du får med indeksen nedover i rekursjonen)

3) translate :: String -> String

Programmer funksjonen **translate :: String -> String**. Denne funksjonen tar inn et ord eller uttrykk i «Oldspeak», og returnerer oversettelsen i «Newspeak». Her skal du bruke ordlisten du har fått utdelt. Dersom ordet ikke finnes i ordlisten, skal funksjonen returnere en tom streng.

Eg:
translate «excellent» = «doubleplusgood»
translate «anti-social tendency» = «ownlife»
translate “haskell” = “”

Ordlisten er **dictionary**, en variabel definert i skjelettprogrammet gitt som vedlegg som du skal fylle ut og levere som din besvarelse. Du må ikke fjerne noe fra denne ordlisten, men du kan legge inn nye oversettelser hvis du ønsker. Hvert element er et par (String, [String]) der listen inneholder ord/fraser i Oldspeak som oversettes til Newspeak ordet som er første elementet i paret. I par der første strengen er tom, dvs. (“”, xs), inneholder listen xs i det andre elementet ord/fraser som ikke har noen oversettelse til NewSpeak.

4) replace :: [(Int,Int)] -> String -> String

Programmer funksjonen **replace :: [(Int,Int)] -> String -> String**. Denne funksjonen skal for hver int-tuple (x,y) bytte ut karakterene i strengen (andre parameter) fra og med indeks x til men ikke med indeks y, med oversettelsen til «Newspeak». Dersom strengen f.o.m. indeks x til men ikke med indeks y ikke har en oversettelse til «Newspeak» skal du erstatte karakterene med ‘*’.

Du kan anta at indeksene i tuplene alltid vil være i rett rekkefølge (x < y) og at de ikke vil være mindre enn 0 eller større enn lengden av strengen. Indekser i forskjellige tupler vil heller ikke overlappe.
Nb: husk at oversettelsen ikke alltid har like mange karakterer som det originale ordet/uttrykket. Indeksene du får inn refererer til den originale strengen. En måte å komme rundt dette på er å sortere indeksene fra størst til minst (dette kan gjøres utvetydig siden indeksene i forskjellige tupler ikke overlapper). Siden du ikke skal bruke funksjoner som ikke er i Prelude, så kan du bruke qsort-funksjonen som vi så på i første ukesoppgave.

Eg:
replace [(11,20)] “Haskell is excellent” = “Haskell is doubleplusgood”
replace [(0,7)] “Haskell is excellent” = “******** is excellent”
replace [(0,7), (11,20)] “Haskell is excellent” = “******** is doubleplusgood”
replace [] “Haskell is excellent” = “Haskell is excellent”

5) toNewspeak :: String -> String

Programmer funksjonen **toNewspeak :: String -> String** som tar inn en streng i «Oldspeak» og returnerer oversettelsen til «Newspeak». Dette innebærer å finne alle forekomster av ordene fra ordlisten, og erstatte dem med det tilsvarende «Newspeak»-ordet, eller '*'-er dersom ordet er i ordlisten, men ikke har en oversettelse. Det aller meste av logikken har du allerede implementert, så denne oppgaven handler i hovedsak om å sette sammen tidligere funksjoner til et sammenhengende program.

6) analytics :: String -> String -> Int

Du ønsker også å analysere tekstene du oversetter for å se hvor mange karakterer som har endret seg. For å forenkle problemet litt bestemmer du deg for å telle antall karakterer fra den originale strengen som ikke er med i den oversatte strengen. Du bryr deg altså ikke om rekkefølgen eller plasseringen til karakterene. F.eks. ville antall endrede karakterer fra «aabb» til «ba» være 2, en 'a' og en 'b'.
Programmer funksjonen **analytics :: String -> String -> Int** som tar inn den originale strengen som første parameter og den oversatte strengen som andre parameter, og returnerer prosentandelen av karakterer som har endret seg fra første til andre streng. Prosenten du returnerer skal være nærmeste heltall tilsvarende:

100 * (<antall endrede karakterer> / <lengde av original streng>)

Eg:
analytics «Julia and Winston Smith think Big Brother is horrible» «***** and ********** think bb is doubleplusungood» = 49
analytics «excellent work» «doubleplusgood work» = 43
analytics «great job» «great job» = 0
analytics «well done» «» = 100

(Hint: **fromIntegral :: (Integral a, Num b) => a -> b** og
round :: (RealFrac a, Integral b) => a -> b kan være nyttige.)

7) main :: String -> (String, Int)

Programmer funksjonen **main :: String -> (String, Int)** som, gitt inputstengen i Oldspeak, returnerer dens oversettelse til Newspeak samt resultatet av analytics.

Skjelett for hele besvarelsen finner du i vedlagt fil **oblig1.hs**. Du skriver først ditt navn i en kommentar øverst, også erstatter du «undefined» ved de enkelte funksjonene med din kode. Klarer du ikke å programmere en funksjon, kan du la den stå uendret med «undefined», og anta at du har en fungerende versjon som kan brukes videre i programmet (hvis funksjonen trengs for å programmere senere funksjoner).

Eg:

main «Julia and Winston Smith think Big Brother is horrible» = («***** and ********** think bb is doubleplusungood», 49)